Europäisches Patentamt	European Patent Office	Office européen des brevets

Bescheinigung Certificate Attestation

Die angehefteten Unterlagen stimmen mit der ursprünglich eingereichten Fassung der auf dem nächsten Blatt bezeichneten europäischen Patentanmeldung überein.	The attached documents are exact copies of the European patent application described on the following page, as originally filed.	Les documents fixés à cette attestation sont conformes à la version initialement déposée de la demande de brevet européen spécifiée à la page suivante.

Patentanmeldung Nr. Patent application No. Demande de brevet n°

02425802.2

Der Präsident des Europäischen Patentamts;
Im Auftrag

For the President of the European Patent Office

Le Président de l'Office européen des brevets
p.o.

R C van Dijk

Anmeldung Nr:
Application no.: 02425802.2
Demande no:

Anmeldetag:
Date of filing: 30.12.02
Date de dépôt:

Anmelder/Applicant(s)/Demandeur(s):

STMicroelectronics S.r.l.
Via C. Olivetti, 2
20041 Agrate Brianza (Milano)
ITALIE

Bezeichnung der Erfindung/Title of the invention/Titre de l'invention:
(Falls die Bezeichnung der Erfindung nicht angegeben ist, siehe Beschreibung.
If no title is shown please refer to the description.
Si aucun titre n'est indiqué se referer à la description.)

Fast page programming architecture and method in a non-volatile memory device with an SPI interface

In Anspruch genommene Priorität(en) / Priority(ies) claimed /Priorité(s) revendiquée(s)
Staat/Tag/Aktenzeichen/State/Date/File no./Pays/Date/Numéro de dépôt:

Internationale Patentklassifikation/International Patent Classification/
Classification internationale des brevets:

G11C/

Am Anmeldetag benannte Vertragstaaten/Contracting states designated at date of filing/Etats contractants désignées lors du dépôt:

AT BE BG CH CY CZ DE DK EE ES FI FR GB GR IE IT LI LU MC NL PT SE SI SK

Titolo: Architettura e metodo per la programmazione page veloce in un dispositivo di memoria non volatile con interfaccia seriale di comunicazione di tipo SPI.

DESCRIZIONE

Campo di applicazione

La presente invenzione riguarda un metodo per la programmazione page veloce in un dispositivo di memoria non volatile con interfaccia seriale di comunicazione di tipo SPI.

L'invenzione riguarda altresì un'architettura di dispositivo elettronico di memoria per l'attuazione del suddetto metodo e del tipo comprendente una matrice di celle di memoria e un'interfaccia seriale di comunicazione di tipo SPI, nonché porzioni circuitali associate alla matrice di celle e preposte alle funzioni di indirizzamento, di decodifica, di lettura, scrittura e cancellazione del contenuto delle celle di memoria.

Più in particolare, l'invenzione riguarda l'implementazione hardware della gestione del bus dei dati e degli indirizzi per effettuare una operazione di programmazione page veloce in una memoria non volatile di tipo flash EEPROM dotata di un protocollo seriale di tipo SPI e la descrizione che segue è fatta con riferimento a questo specifico campo di applicazione con lo scopo di semplificarne l'esposizione.

Arte nota

Com'è noto ai tecnici del ramo di questo specifico settore tecnico, attualmente il mercato delle memorie non volatili è suddiviso in quattro grandi categorie:

- EEPROM programmabili per byte;
- EEPROM programmabili per page (dove una page include una pluralità di bytes);
- FLASH programmabili per byte; e,
- FLASH programmabili per page.

A fronte di una maggiore semplicità strutturale delle memorie FLASH rispetto a quella delle EEPROM, con vantaggi in termini di area di silicio

occupata e di costi, nelle Flash risulta piuttosto gravoso programmare molte parole di memoria (word) contemporaneamente in termini di potenza dissipata.

Per superare questo limite è stata escogitata la programmazione page e si è provveduto ad inserire all'interno della memoria Flash, con interfaccia di comunicazione SPI, una memoria SRAM usata come buffer in cui memorizzare le word da programmare in matrice.

Se si immagina che una matrice di memoria Flash è suddivisa in settori comprendenti pagine di parole di memoria (word) da 8 bit a parola, ogni parola è identificata nei bit meno significativi, A7-A0, della codifica di bit d'indirizzo che vanno da A23 ad A0.

La memoria SRAM è indirizzata utilizzando i suddetti bit meno significativi in modo da avere in questo registro buffer la copia immagine delle word delle pagine presenti in matrice e indirizzate dai bit più significativi, vale a dire quelli che vanno da A23 ad A8.

In questo genere di memorie, la programmazione in modalità page viene generalmente effettuata immagazzinando tutte le word nella SRAM e successivamente andando a leggerne il contenuto per singola word. Se tale contenuto risulta diverso da "FF" (in una Flash i bit a valore logico "1" stanno ad indicare una cella cancellata), sarà immagazzinato nella corrispondende word della pagina del settore di matrice.

Vediamo ora brevemente la modalità di funzionamento di una memoria flash con protocollo seriale di comunicazione SPI durante l'operazione di scrittura.

Per effettuare una operazione di scrittura in una memoria Flash, indipendentemente dal protocollo usato, è necessario dapprima togliere la protezione contro la scrittura del settore di matrice da programmare, indirizzando un opportuno registro di protezione. Successivamente, occorre dare un comando di scrittura, che sarà decodificato da una specifica unità preposta (Command User Interface). Questo comando serve ad abilitare un algoritmo interno di programmazione e successivamente passare l'indirizzo di matrice e il dato o i dati da programmare.

Queste tre fasi possono essere attuate secondo il protocollo SPI controllato da una macchina a stati.

Una programmazione di tipo Page, sempre in modalità SPI, comporta l'esecuzione di una opportuna sequenza di istruzioni:

- la prima, denominata WREN (Write Enable), serve ad abilitare il dispositivo di memoria ad ogni tipo di scrittura, come schematicamente illustrato nella sequenza di segnali di figura 1;
- la seconda, denominata WRSR (Write Status Register), permette di togliere la protezione all'area di memoria indirizzata, come schematicamente illustrato nella sequenza di segnali di figura 2;
- la terza, denominata PP (Page Program), permette di passare il comando di Page Programmazione, l'indirizzo e i dati da programmare, come schematicamente illustrato nella sequenza di segnali di figura 3;

Nelle figure 1, 2 e 3, il segnale S_N rappresenta il Chip Select, sul cui fronte di discesa viene accenso il dispositivo di memoria, ovvero vengono abilitati tutti i buffers d'ingresso.

Il segnale C rappresenta invece il segnale di temporizzaizone o di Clock, che permette di sincronizzare le varie fasi del protocollo SPI.

Il segnale D rappresenta il pin d'ingresso, attraverso il quale vengono passati istruzioni, indirizzi e dati; mentr eil segnale Q rappresenta il pin d'uscita, attraverso il quale vengono portati verso l'esterno i dati letti.

Tenuto conto che una memoria di tipo FLASH può programmare un byte alla volta, per implementare la page programmazione risulta evidente che è necessario immagazzinare in una opportuna struttura le sequenze di dati che devono essere programmati.

In particolare, supponendo di voler programmare fino a 256 bytes, si dovrà dotare il dispositivo di una memoria di tipo volatile, come la suddetta SRAM, in grado di immagazzinare fino a 2048 bit.

Tenuto conto dell'elevato numero di bit da immagazzinare, ne consegue che l'ottimizzazione della struttura risulta un fattore determinante per quanto concerne l'occupazione di area su silicio.

Come già detto in precedenza, il modo più semplice per implementare la page programmazione è quello di adoperare un banco di memoria tampone di tipo SRAM (Static Random Access Memory). Ovviamente un banco di memoria di questo tipo presenta una complessità eccessiva per il compito che deve svolgere. Ad esempio risulta inutile avere un accesso casuale alla memoria, il che comporta l'uso di una circuiteria di decodifica di riga e di colonna, se poi alla fine su una memoria FLASH si programma ad indirizzi successivi.

La struttura ottima sarebbe quella completamente seriale, ad esempio come quella di un registro a scorrimento o SHIFT-REGISTER, la quale soffrirebbe però di eccessivi assorbimenti nel caso in cui più elementi latch commutassero simultaneamente. Inoltre uno SHIFT-REGISTER presuppone l'uso di un Flip-Flop Master-Slave e quindi di un numero notevole di transistori per ciascun bit da memorizzare.

Il problema tecnico che sta alla base della presente invenzione è quello di escogitare un'architettura di dispositivo di memoria ed un relativo metodo per la programmazione page veloce, in particolare per un dispositivo di memoria non volatile dotato di un'interfaccia seriale di comunicazione di tipo SPI, aventi rispettive caratteristiche tali da consentire l'"mpiego di una unità di memoria tampone di modesta complessità e costo di realizzazione contenuto.

Sommario dell'invenzione

L'idea di soluzione che sta alla base della presente invenzione è quella di prevedere un banco di memoria tampone incorporato nel dispositivo di memoria e destinato a memorizzare e prelevare dati durante la page programmazione in modalità pseudo-seriale consentendo il latching dei dati un bit alla volta ed un successivo prelievo di almeno due bytes per volta.

Sulla base di questa idea di soluzione, il problema tecnico è risolto da un'architettura del tipo precedentemente indicato e definito dalla rivendicazione 1 e seguenti.

Il problema tecnico è risolto inoltre da un metodo di page programmazione definito nelle rivendicazioni 8 e seguenti.

Le caratteristiche ed I vantaggi del metodo e dell'architettura secondo l'invenzione risulteranno dalla descrizione, fatta qui di seguito di un esempio di attuazione dato a titolo indicativo e non limitativo con riferimento ai disegni allegati.

Breve descrizione dei disegni

- la figura 1 mostra su un diagramma a medesima base temporale una seqenza di segnali per abilitare un dispositivo di memoria di tipo noto ad ogni tipo di scrittura;
- la figura 2 mostra su un diagramma a medesima base temporale una seqenza di segnali per togliere la protezione ad un'area di memoria indirizzata da programmare in modalità page in un dispositivo di tipo noto;
- la figura 3 mostra su un diagramma a medesima base temporale una seqenza di segnali per passare il comando di Page Programmazione, l'indirizzo e i dati da programmare in un dispositivo di memoria convenzionale;
- la figura 4 mostra una vista schematica dell'architetura di un dispositivo elettronico di memoria non volatile realizzato in accordo con la presente invenzione;
- la figura 5 mostra una vista schematica di un particolare del dispositivo di figura 4;
- la figura 6 mostra uno schema di dettaglio circuitale di una porzione del dispositivo di figura 4;
- le figure 7 e 8 mostrano su rispettivi diagrammi a medesima base temporale un insieme di segnali significativi di una simulazione di Page Programmazione veloce nel dispositivo di memoria secondo l'invenzione;
- la figura 9 mostra una vista schematica che illusta la modalità di latching dei dati un bit alla volta ed un successivo prelievo a due byte (sedici bit) alla volta nel dispositivo secondo l'invenzione.

Descrizione dettagliata

Con riferimento a tali figure, e in particolare all'esempio di figura 4, con 1 è globalmente e schematicamente indicata l'architettura di un dispositivo elettronico di memoria non volatile integrato monoliticamente su semiconduttore e realizzato in accordo con la presente invenzione.

Il dispositivo 1 è strutturato per consentire una programmazione page veloce ed è dotato di un'interfaccia 2 seriale di comunicazione di tipo SPI.

Per dispositivo di memoria si intende un qualunque sistema elettronico monolitico incorporante una matrice 3 di celle di memoria, organizzate in righe e colonne, e porzioni circuitali associate alla matrice di celle e preposte alle funzioni di indirizzamento, di decodifica, di lettura, scrittura e cancellazione del contenuto delle celle di memoria.

Un dispositivo di questo genere può essere ad esempio un chip di memoria integrato su semiconduttore e del tipo Flash EEPROM non volatile suddivisa in settori e cancellabile elettricamente.

Come noto, ciascuna cella di memoria comprende un transistore a floating gate con terminali di source, drain e control gate.

Tra le porzioni circuitali associate alla matrice di celle è prevista la presenza di una porzione circuitale di decodifica di riga associata a ciascun settore ed alimentata da specifiche tensioni positive e negative generate ad esempio internamente al circuito integrato di memoria mediante survoltori o pompe di carica e regolate tramite relativi regolatori di tensione.

L'interfaccia 2 seriale di comunicazione che supporta una modalità di funzionamento seriale di tipo SPI per applicazioni a ridotto numero di pin. La programmazione di tipo page viene implementata in diverse applicazioni come microprocessori, schede audio, schede grafiche, ecc.

Vantaggiosamente, secondo l'invenzione, il dispositivo 1 incorpora un banco 5 di memoria tampone da utilizzare durante la fase di page programmazione.

Rispetto alle soluzioni di tipo noto, la struttura del banco 5 di memoria è stata particolarmente semplificata facendo in modo che la

memorizzazione dei dati in esso e il successivo prelievo, possa avvenire con una modalità pseudo-seriale.

Inoltre, è possibile sovrascrivere i dati memorizzati nel banco 5 di memoria per le successive fasi di page programmazione senza preoccuparsi del loro reset. Tutto ciò si traduce in un risparmio dell'ordine di decine di migliaia di transistori normalmente necessari per la selezione, cancellazione, etc.

L'architettura del dispositivo 1 comprende una pluralità di blocchi funzionali che vengono utilizzati durante l'operazione di scrittura.

Nella figura 4 è mostrato lo schema a blocchi dei circuiti interessati all'operazione di page programmazione, mentre nella figura 5 è mostrato in maggiore dettaglio il banco 5 di memoria DQLATCHES per l'immagazzinamento e la gestione dei dati della page programmare.

Come si può apprezzare dalla figura 4, l'architettura 1 prevede un primo ingresso D, attraverso il quale i dati, gli indirizzi e le istruzioni, sono passati ad una macchina a stati 6 che rappresenta l'interfaccia tra l'utente e il dispositivo 1 di memoria; sono inoltre trasferiti un segnale di WE, generato dal blocco DQLATCHES, e un segnale LOAD_DATA, proveniente da un'unità di elaborazione CUI (Command Users Interface).

Il segnale di WE ha la doppia funzione di sincronizzare i dati, attraverso il percorso che va dalla macchina a stati 6 alla matrice 3, e dalla macchina a stati 6 alla CUI, mentre quello di LOAD_DATA ha la funzione di caricare i dati nei Program Load 8 da cui, successivamente, passeranno nella matrice 3, durante l'algoritmo di programmazione.

Vediamo di descrivere più dettagliatamente il percorso dei dati durante l'operazione di programmazione dal piedino d'ingresso D alla matrice 3 e i blocchi interessati.

L'istruzione di Page Programmazione prevede il passaggio, attraverso l'ingresso D, del comando di Page Programming, dell'indirizzo e, in successione, di tutti i bytes da programmare.

L'indirizzo, preferibilmente a 24 bit, viene memorizzato nel blocco ADDLATCH della macchina a stati 6 e da qui viene successivamente

trasferito all'ADDRESS COUNTER 9 del dispositivo di memoria 1. I dati, invece, sono memorizzati nel banco 5 tampone DQLATCHES e da qui vengono trasferiti poi nei Program Load 8 del dispositivo 1.

Il comando di programmazione, proveniente dal DBUS, è decodificato dall'unità CUI che abilita l'algoritmo interno di programmazione.

Una volta memorizzati temporaneamente tutti i dati nel banco 5 di memoria tampone, sul fronte di salita del segnale Chip Select (S_N) vengono caricati i primi due bytes nei Program Load 8, attraverso il blocco 7 DATAL_IO che gestisce il trasferimento dei dati dai DQLATCHED<15:0> al bus interno DBUS<15:0>, ed inizia l'algoritmo di programmazione che provvede a programmare in matrice 3 i due bytes suddetti.

A fine algoritmo, sul fronte di discesa del segnale MODIFY proveniente dall'unità CUI, segnale che resta alto per tutta la durata dell'algoritmo di programmazione, viene incrementato l'indirizzo memorizzato nell'ADDLATCH. L'incremento è di due unità dal momento che si è scelto di programmare due bytes per volta.

A questo punto viene generato un nuovo segnale di LOAD_DATA e vengono caricati nei Program Load 8 i successivi due dati latchati nel banco 5 DQLATCHES.

Anche il segnale MODIFY viene innalzato e inizia nuovamente l'algoritmo di programmazione. Ad ogni fine algoritmo viene fatto il confronto tra le uscite dei due contatori, C1<8:0> e C2<8:0>.

Nel momento in cui sarà terminata la programmazione di tutti i bytes latchati (C1=C2), il banco di memoria DQLATCHES provvederà a fornire un segnale di fine programmazione che scatenerà un reset della macchina a stati 6 e dell'interfaccia 2, nonché dell'unità CUI.

Nella figura 5 è illustrata in maggiore dettaglio la struttura interna del banco 5 di memoria tampone DQLATCHES.

Sono individuabili dei blocchi fondamentali: un blocco 10 DATALATCHES, comprendente una batteria di 2048 latches, come quelli rappresentati schematicamente in figura 6, che contiene i dati da programmare; i blocchi 11, 12, 13 denominati anche BANKIN_GEN,

BANK_GEN e DQDEMUX, che consentono il latching un bit alla volta dei dati e il successivo prelievo due bytes per volta.

Vi sono anche: due blocchi 15, 16 denominati COUNTER, che consentono di tenere conto del numero di bytes da programmare e del numero di bytes già programmati in matrice 3 ad un determinato istante dell'operazione di Page Programmazione, un blocco 14 comparatore COMP, suscettibile di generare un impulso di reset nel momento in cui il numero di bytes programmati (C2) eguaglia quello di bytes da programmare (C1) ed una logica 20 di controllo LOGIC CONTROL.

Questa struttura logica 20 tiene conto del numero pari o dispari di bytes da programmare e dell'indirizzo iniziale e fornisce opportuni segnali di controllo al blocco 7 DATAL_IO di figura 4 per il corretto caricamento dei dati nei Program Load 8.

La struttura adottata per il latching del singolo bit nella presente proposta di brevetto è quella di un semplice latch con due segnali di abilitazione: BANKIN e B, e un ingresso dati (D). In figura 6 è mostrata schematicamente che consente di ottenere un layout molto compatto e un risparmio notevole di area su silicio, data la sua semplicità.

Per apprezzare appieno il funzionamento dei blocchi precedentemente illustrati, si può fare riferimento agli esempi delle figure 7 e 8 che riportano l'andamento dei segnali in una simulazione di Page Programmazione effettuata a 50 MHz. Si suppone di programmare tre bytes.

In Figura 7 si nota che in seguito all'istruzione di WREN si alza il segnale WE_LATCH che abilita la memoria 3 ad ogni tipo di scrittura.

A questo punto, passando l'istruzione di PP si genera un impulso di WE che consente di far decodificare all'unità CUI il comando presente su DBUS e, di conseguenza, di far alzare il segnale di PAGE_PROG.

Tale segnale PAGE_PROG abiliterà tutta la logica descritta in precedenza con riferimento all figure 4 e 5.

Le fasi successive sono quelle di passaggio dei 24 bit dell'indirizzo, che viene memorizzato negli ADDLATCHED<23:0>, e di passaggio dei dati

da programmare che sono memorizzati nei DATALATCHED<0:2047>, come da figura 8.

La struttura di memoria utilizzata si può pensare come suddivisa in 256 sottobanchi di otto latch ciascuno.

Ciascun sottobanco è abilitato da uno dei segnali BANK<0:255>, mentre i singoli latch sono abilitati da uno dei segnali BANK_IN<0:7>.

Per memorizzare quindi l'n-esimo byte occorre far alzare il segnale BANK<n-1> e, in successione, i segnali BANK_IN<0:7>.

In sostanza, è ome se ci fosse un doppio multiplexing dei bytes e dei bit all'interno dei bytes. Questa situazione è schematizzata in Figura 9.

Una volta latchati tutti i bytes, sul fronte di salita di S_N i primi due bytes vengono caricati nei DQLATCHED<15:0>, nasce un segnale di WE, che fa passare i dati su DBUS<15:0>, mentre il successivo segnale di LOAD_DATA carica tali dati nei Program Load 8. E' da notare che il contatore C1<8:0> contiene il numero di bytes da programmare.

I segnali di controllo FORCE_DBUS, FORCE_DBUS_SHIFT e BIT_A0 gestiscono il caricamento corretto dei dati sul bus DBUS e da questo nei Program Load (TO_BE_PROG<31:0>). Si noti che i Program Load contengono i negati dei dati da programmare.

A questo punto parte l'algoritmo di programmazione vero e proprio, ovvero si alza il segnale MODIFY.

Dal momento che il numero di bytes da programmare è dispari, il contatore C2<8:0> segna inizialmente 1(h). In effetti, se fosse stato pari il valore iniziale di C2 sarebbe stato 2(h).

A fine algoritmo, vale a dire una volta programmati i primi due bytes, sul fronte di discesa di MODIFY, essendo C1 e C2 diversi, quest'ultimo viene incrementato di due, segnando 3(h), o 4(h) nel caso pari.

Sullo stesso fronte di discesa nasce un impulso (INC_ADD) che incrementa di due unita l'indirizzo. Inoltre nasce un nuovo WE che carica il terzo byte su DBUS e da qui, tramite il segnale di LOAD_DATA, nei Program Load 8.

Si alza nuovamente il segnale MODIFY, ovvero inizia per la seconda volta l'algoritmo di programmazione.

Alla fine di questa fase di programmazione, il terzo e ultimo byte, essendo C1=C2=3(h), nasce un segnale di reset che resette la macchina a stati 6.

Il clock che gestisce la generazione dei segnali BANK<0:255>, CK_BANK, coincide, durante la fase di latching dei bytes nei DATALATCHED, col segnale BANK_IN<7>, che permette il latching dell'ultimo bit di ciascun byte, e durante tale fase ogni BANK individua un byte da latchare, mentre durante la fase di programmazione vera e propria coincide col segnale MODIFY, e durante tale fase ogni BANK individua due bytes da programmare.

Per esempio, il BANK<0> individua nella prima fase il byte 0 da latchare, mentre nella seconda fase individua i bytes 0 e 1 latchati da programmare.

Pertanto, nella fase di programmazione vera e propria, i BANK sono 128 (BANK<0:127>), ed ogni qualvolta si alza un BANK, due nuovi bytes, latchati nel blocco DQLATCHES, vengono caricati sui DQLATCHED<15:0> e da qui nei Program Load 8.

L'architettura ed il metodo secondo l'invenzione risolvono il problema tecnico e conseguono diversi vantaggi qui di seguito evidenziati.

In primo luogo è stata escogitata una struttura logica che permette di combinare al meglio l'ingresso dati con il bus interno a sedici bit.

L'innovazione introdotta consiste nel metodo adottato per effettuare la programmazione page e nella soluzione circuitale utilizzata per implementare quest'ultimo.

La logica utilizzata, unitamente all'impiego di un latch a singolo bit molto semplice, consente di rendere estremamente compatto il layout circuitale del dispositivo di memoria. Ciò comporta ovviamente notevoli vantaggi in termini di area di silicio occupata e, di conseguenza, in termini di costi.

Inoltre l'architettura proposta, oltre a permettere di lavorare a frequenze elevate fino a 50 MHz), prevede la possibilità di programmare due bytes alla volta, dimezzando il tempo di programmazione complessivo. In più con la semplice aggiunta di un pin esterno di Vpp=12V (Program Supply Voltage), si potrebbero programmare benissimo quattro bytes alla volta, riducendo ulteriormente il tempo suddetto.

RIVENDICAZIONI

1. Architettura di dispositivo (1) elettronico di memoria non volatile per la page programmazione veloce, del tipo comprendente una matrice (3) di celle di memoria e un'interfaccia (2) seriale di comunicazione di tipo SPI, nonché porzioni circuitali associate alla matrice (3) di celle e preposte alle funzioni di indirizzamento, di decodifica, di lettura, scrittura e cancellazione del contenuto delle celle di memoria, caratterizzata dal fatto di comprendere un banco (5) di memoria tampone per memorizzare e prelevare dati durante la page programmazione in modalità pseudo-seriale.

2. Architettura secondo la rivendicazione 1, caratterizzata dal fatto che detto banco (5) di memoria tampone è incorporato in una macchina a stati (6) ricevente in ingresso dati, indirizzi e istruzioni per accedere alla memoria (3) e produce un segnale (WE) di sincronizzazione dati attraverso un percorso che va dalla macchina a stati (6) alla matrice di memoria (3).

3. Architettura secondo la rivendicazione 1, caratterizzata dal fatto che detto banco (5) di memoria tampone comprende una batteria di latches contenente i dati da programmare e blocchi (11, 12, 13) di memoria che consentono il latching un bit alla volta dei dati e il successivo prelievo di almeno due bytes per volta.

4. Architettura secondo la rivendicazione 3, caratterizzata dal fatto che detto banco (5) di memoria tampone comprende anche due blocchi (15, 16) contatori per tenere conto del numero di bytes da programmare e del numero di bytes già programmati in matrice (3) ad ogni determinato istante dell'operazione di Page Programmazione.

5. Architettura secondo la rivendicazione 4, caratterizzata dal fatto di comprendere un blocco (14) comparatore collegato a valle di detti contatori (15, 16) e suscettibile di generare un impulso di reset nel momento in cui il numero di bytes programmati (C2) eguaglia quello di bytes da programmare (C1).

6. Architettura secondo la rivendicazione 3, caratterizzata dal fatto

di comprendere ulteriormente una logica (20) di controllo per tenere conto del numero pari o dispari di bytes da programmare e dell'indirizzo iniziale in cui avviare la page programmazione, nonché fornire segnali di controllo ad un blocco (7) preposto al caricamento dati nei Program Load (8).

7. Architettura secondo la rivendicazione 3, caratterizzata dal fatto il latching del singolo bit è effettuato da un semplice latch a due segnali (BANKIN, B) di abilitazione e un ingresso dati (D).

8. Metodo per effettuare una page programmazione in dispositivi elettronici di memoria non volatile dotati di una matrice (3) di celle di memoria e un'interfaccia (2) seriale di comunicazione di tipo SPI, nonché porzioni circuitali associate alla matrice (3) di celle e preposte alle funzioni di indirizzamento, di decodifica, di lettura, scrittura e cancellazione del contenuto delle celle di memoria, caratterizzato dal fatto di prevedere un banco (5) di memoria tampone per memorizzare e prelevare dati durante la page programmazione in modalità pseudo-seriale attraverso detta interfaccia (2).

9. Metodo secondo la rivendicazione 8, caratterizzato dal fatto che in detto banco (5) di memoria tampone avviene un latching dei dati un bit alla volta e il successivo prelievo di almeno due bytes per volta.

10. Metodo secondo la rivendicazione 8, caratterizzato dal fatto che un'istruzione di Page Programmazione prevede il passaggio attraverso un ingresso (D) di detta interfaccia (2) del comando di Page Programming, dell'indirizzo di memoria e, in successione, di tutti i bytes dai dati da programmare.

11. Metodo secondo la rivendicazione 8, caratterizzato dal fatto che i dati contenuti nel banco (5) di memoria sono sovrascritti all'avvio delle successive fasi di page programmazione senza effettuare un reset del banco stesso.

12. Metodo secondo la rivendicazione 8, caratterizzato dal fatto di prevedere anche un conteggio del numero di bytes da

programmare e del numero di bytes già programmati in matrice (3) ad ogni determinato istante dell'operazione di Page Programmazione.

RIASSUNTO

L'invenzione rigurada un'architettura circuitale ed un metodo per effettuare una page programmazione in dispositivi elettronici di memoria non volatile dotati di una matrice (3) di celle di memoria e un'interfaccia (2) seriale di comunicazione di tipo SPI, nonché porzioni circuitali associate alla matrice (3) di celle e preposte alle funzioni di indirizzamento, di decodifica, di lettura, scrittura e cancellazione del contenuto delle celle di memoria. Vantaggiosamente, è previsto un banco (5) di memoria tampone per memorizzare e prelevare dati durante la page programmazione in modalità pseudo-seriale attraverso detta interfaccia (2). Il latching dei dati viene eseguito un bit alla volta e il successivo prelievo avviene invece con almeno due bytes per volta.

(Fig. 4)

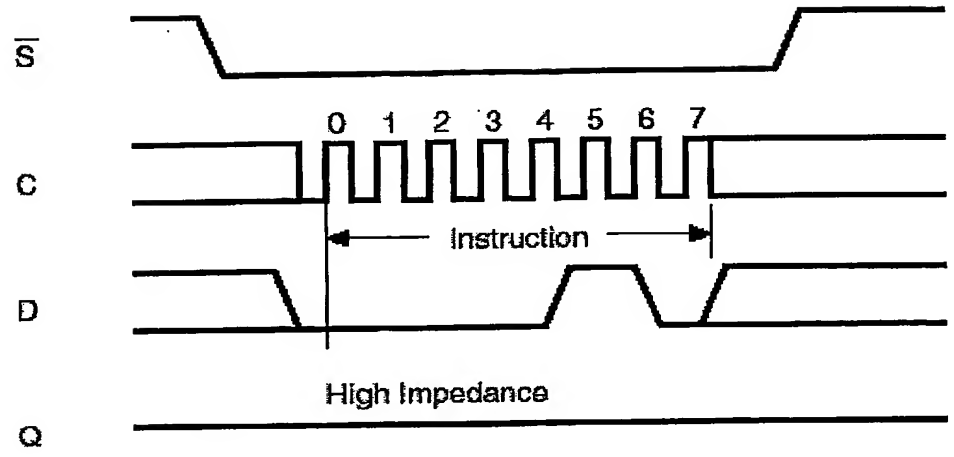
S
0 1 2 3 4 5 6 7
C
Instruction
D
High Impedance
Q

Fig.1

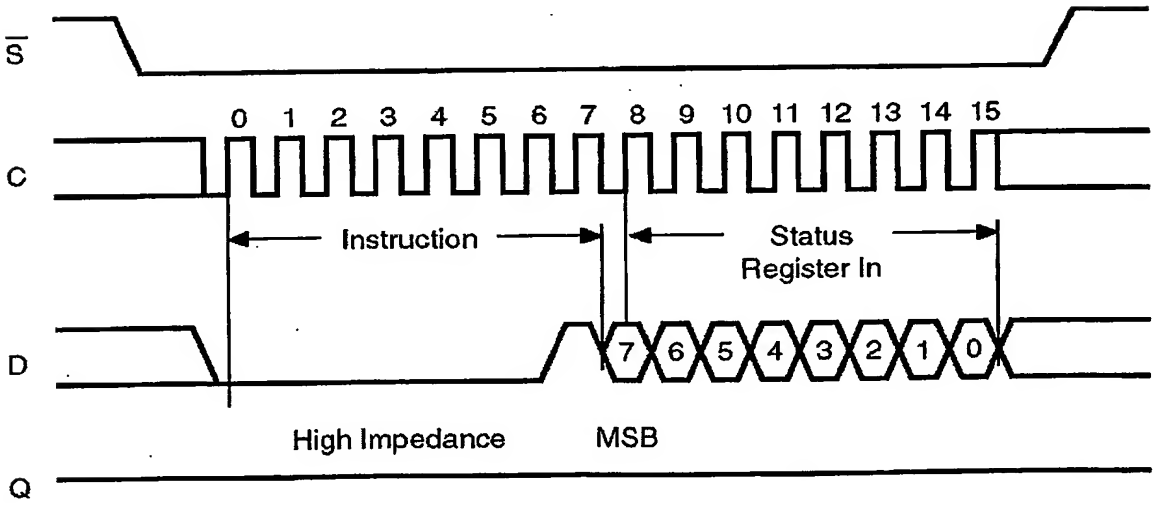
S
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
C
Instruction
Status
Register In
D
7 6 5 4 3 2 1 0
High Impedance
MSB
Q

Fig.2

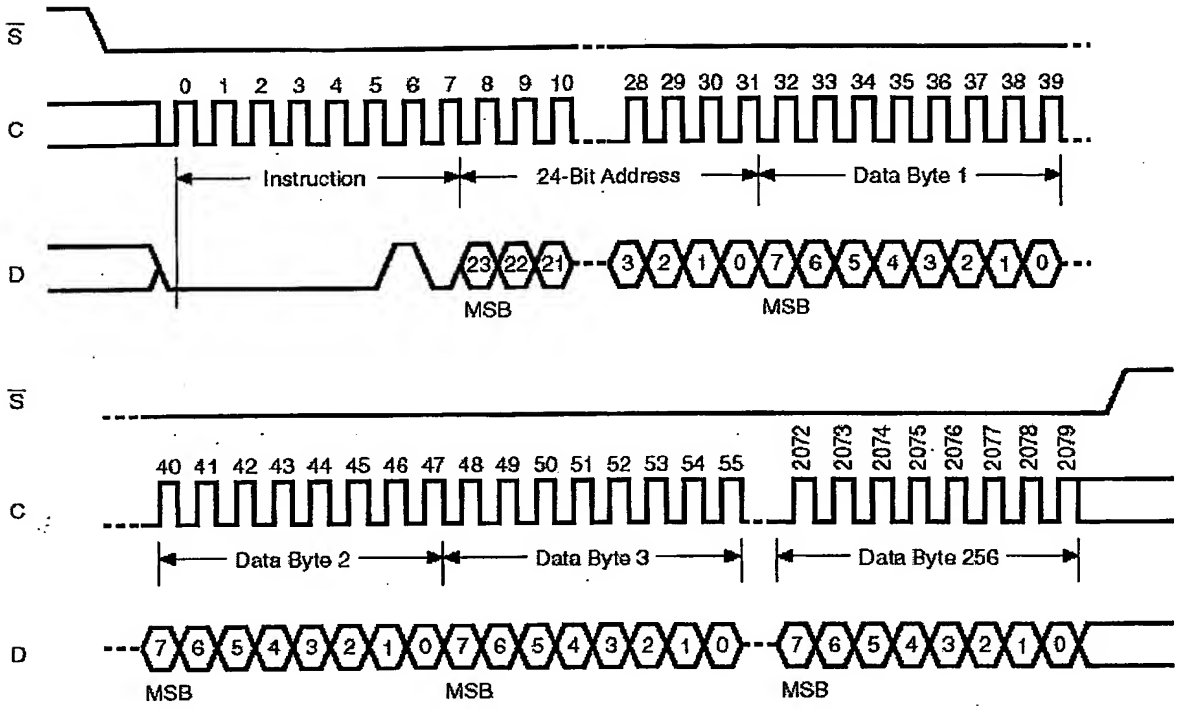
S
C
D
0 1 2 3 4 5 6 7 8 9 10 28 29 30 31 32 33 34 35 36 37 38 39
Instruction
24-Bit Address
Data Byte 1
23 22 21 3 2 1 0 7 6 5 4 3 2 1 0
MSB
MSB
40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 2072 2073 2074 2075 2076 2077 2078 2079
Data Byte 2
Data Byte 3
Data Byte 256
7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0
MSB
MSB
MSB

Fig.3

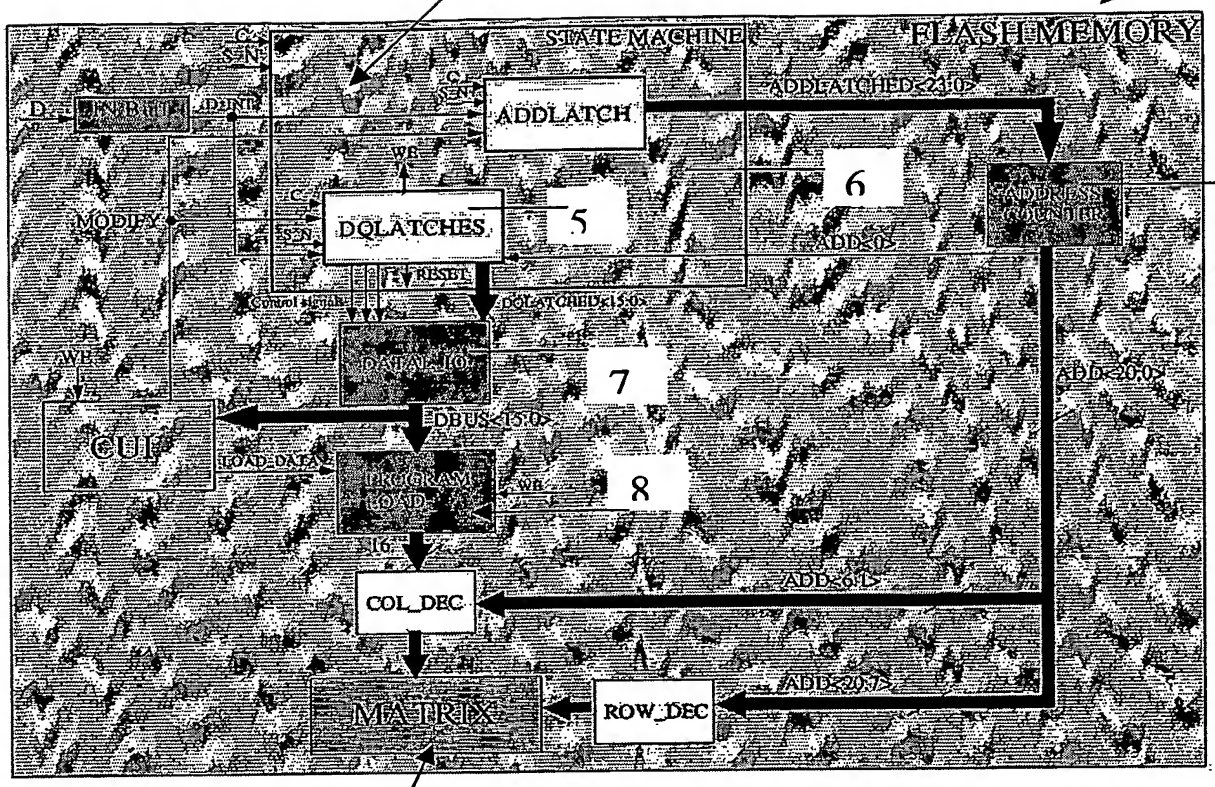
1
2
FLASH MEMORY
STATE MACHINE
ADDLATCH
ADDLATCHED<23:0>
6
9
ADDRESS COUNTER
ADD<0:5>
DQLATCHES
5
RESET
DQLATCHED<15:0>
DATA I/O
7
ADD<20:0>
MODIFY
WR
CUI
DBUS<15:0>
LOAD_DATA
PROGRAM LOAD
8
WR
16
COL_DEC
ADD<6:15>
MATRIX
ROW_DEC
ADD<20:7>
3

Fig.4

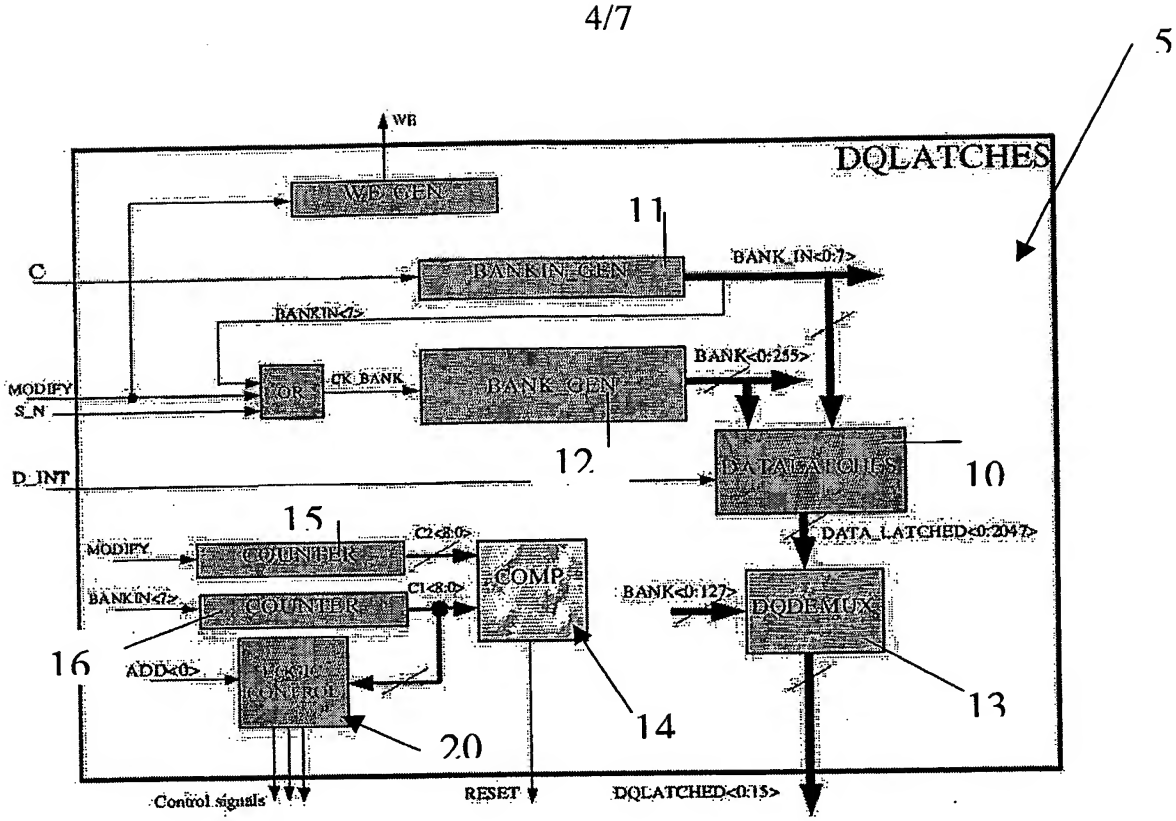
5
WB
DQLATCHES
WB_GEN
11
C
BANKIN_GEN
BANK_IN<0:7>
BANKIN<7>
MODIFY
S_N
XOR
CK_BANK
BANK_GEN
BANK<0:255>
12
D_INT
DATALATCHES
10
DATA_LATCHED<0:2047>
15
MODIFY
COUNTER
C2<8:0>
COMP
BANKIN<7>
COUNTER
C1<8:0>
BANK<0:127>
DQDEMUX
16
ADD<0>
CLOCK CONTROL
13
14
20
Control signals
RESET
DQLATCHED<0:15>

Fig.5

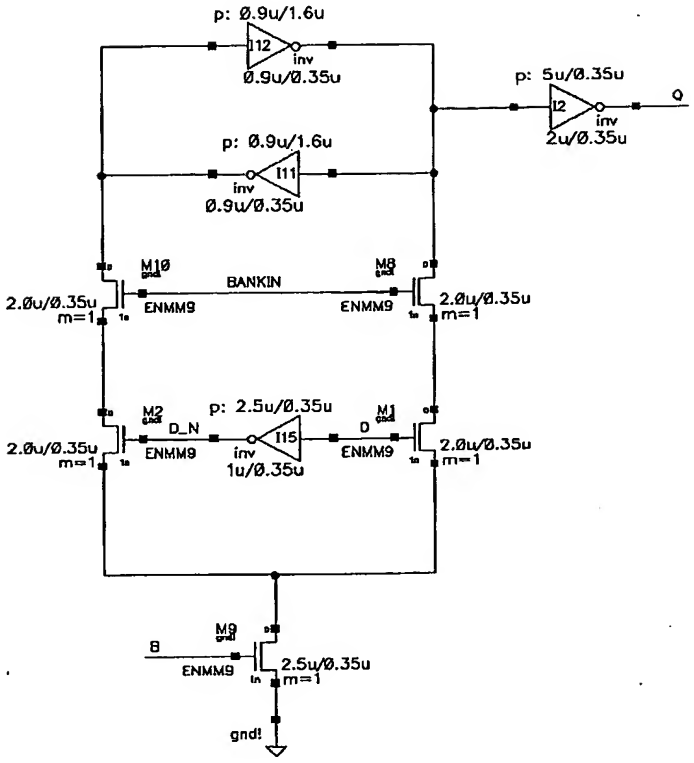
p: Ø.9u/1.6u
I12
inv
Ø.9u/Ø.35u
p: 5u/Ø.35u
I2
inv
2u/Ø.35u
Q
p: Ø.9u/1.6u
I11
inv
Ø.9u/Ø.35u
M10
BANKIN
M8
2.Øu/Ø.35u
m=1
ENMM9
ENMM9
2.Øu/Ø.35u
m=1
M2
p: 2.5u/Ø.35u
M1
D_N
D
2.Øu/Ø.35u
m=1
ENMM9
I15
ENMM9
2.Øu/Ø.35u
m=1
inv
1u/Ø.35u
M9
B
ENMM9
2.5u/Ø.35u
m=1
gnd!

Fig.6

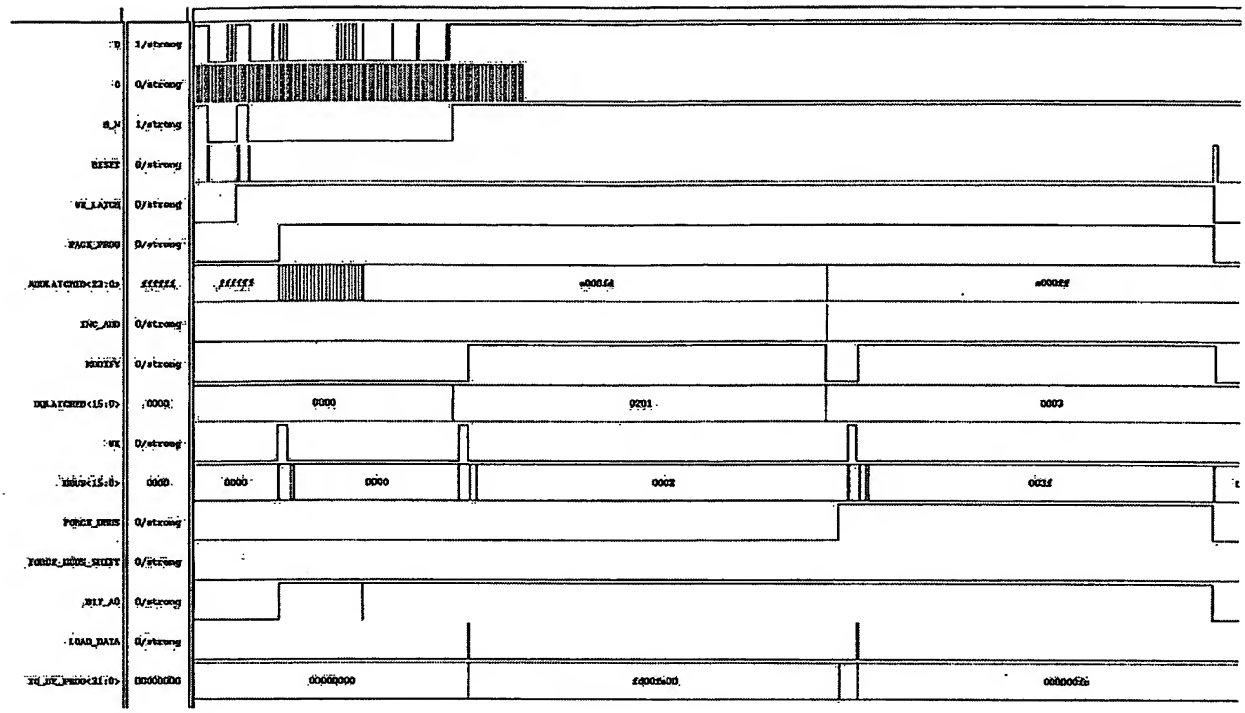

Fig.7

Fig.8

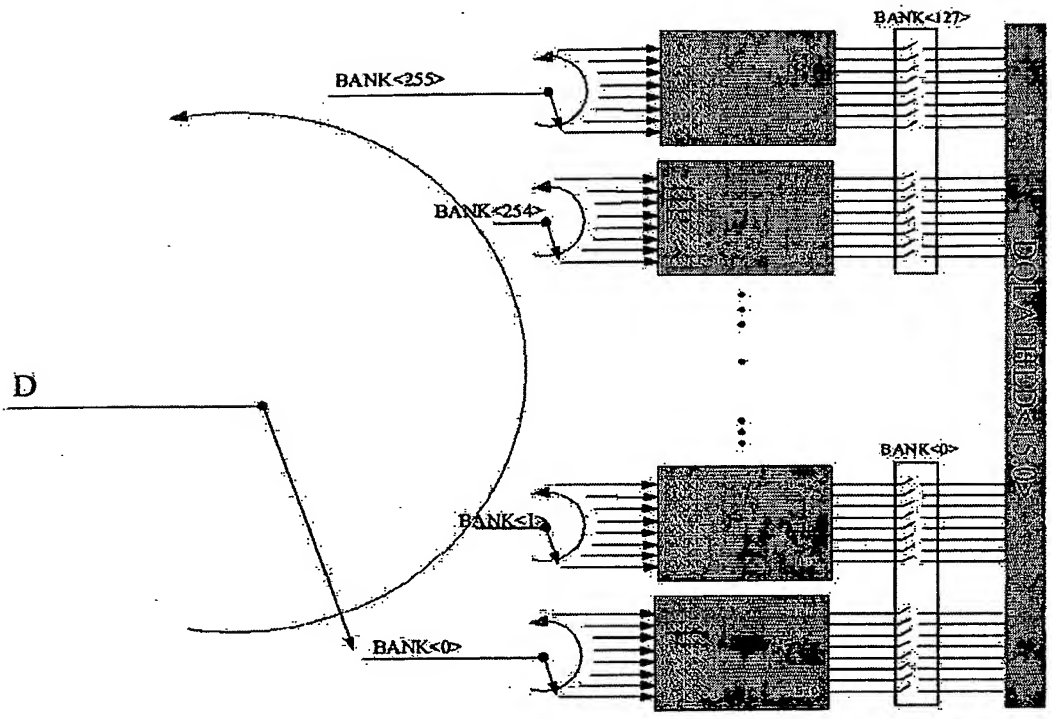

Fig.9